commodore
PET USERS CLUB NEWSLETTER

Issue No. 7

Commodore News

By the time you have received this newsletter, most of Commodore Systems will have moved to new premises in Slough. This marks the completion of the trade operation we have been setting up over the last few months. With over 120 Pet dealers in this country, there is now a well established network of centres for Pet activity - including software, hardware and service. Our new headquarters at Slough are purely for trade enquiries and services to dealers.

For the time being our premises at 360 Euston Road, London will remain open as a central showroom. Articles and news for this newsletter should still be sent to this address.

FLOPPY DISCS

First the good news, most dealers have had a number of floppy discs delivered . If you would like to know more about the operation of this unit ask your dealer for a demonstration.

Now for the not so good news; owing to the world shortage of floppy disc drives we have been forced to increase the price of the 2040 disc drive to £795 + VAT. We still think that this unit offers outstanding value for money.

IEEE CABLE-CONNECTORS

Because different users will have different systems and different requirements we have decided to sell the IEEE connectors as separate items. These will be available as follows:

PET to IEEE connector	:	£20 + VAT
IEEE to IEEE connector	:	£25 + VAT

It is important that you specify on any order which connector is required. For a full PET, printer, floppy disc system, you will need to purchase one of each connector.

PRINTERS

In order to speed up the production of printers we have set up a completely new production line in Japan to cope with European sales. As most of the demand has been for the tractor feed printer, the plant will initially only be producing this type. The electronics and mechanics will remain the same as the 2022 but as a different housing has been necessitated, this printer will be designated the 3022.

ENDORSEMENT SCHEME

For the past few months we have been keeping a directory of all the services, peripherals and programs available for the PET from independent sources - though without making specific recommendations.

In view of the wide range of products and services available we have set up a new 'Endorsement Scheme' which works as follows:

Any product can be submitted for the Endorsement scheme and someone inside Commodore has been appointed to deal exclusively with this area. We then test the product for its compatibility or suitability and to see how reliable it is. If it is found to be of a suitably high quality then it is entered on our list of endorsed products. The manufacturer or distributor is then allowed to market the product using the phrase: "Recommended by Commodore" and with the Commodore logo. In this way, the user will be given some guide as to the quality of the product or service offered.

Several items have been submitted for consideration and we will be listing some of the new entries in this newsletter. First off the line however are two new business packages from PETACT - specifically Purchase Accounting and Sales Accounting. These are professionally written systems designed for use on a 32K PET (Floppy disc is not necessary) plus a printer. Specially printed stationery is included to produce Debtors/Creditors control lists, invoices, statements, cheques and ledger records etc. The price of these programs is not cheap, but includes a full day's training on use of the system. For further details contact:

Applied Computer Techniques (Holdings) Ltd.,
MicroAct Ltd.,
5/6 Vicarage Road,
Edgebaston,
Birmingham B15 3ES.

Telephone : 021-454-5341

Or contact your local dealer.

Stop Press

latest on the endorsement scheme is a field service contract available nationally or through local dealers. Terms are 10% of hardware value and the service call-out time is 24 hours.

For further details contact:

D.D.T Maintenence Ltd.,
58-60 Northfield Road,
Kings Norton
Birmingham B30 1JH. 021-459-5957

Editorial

First of all, a big thank you to the scores of members who returned the questionnaire in issue No.6. It has taken me some time to wade through your answers, but I feel that the exercise was a useful one in getting feedback from readers. As promised, your comments have not fallen on deaf ears and so I have outlined some of the more common answers (and the proposed course of action) below:

1) Readers found that the most useful part of the newsletter was the programming section and several members asked if it could be lengthened. In response to this I have tried to include as much programming as possible in this issue. I am very dependent however on your sending in articles for publishing - so please keep them coming.

2) Of the possible new sections suggested, by far the most common request was for a Machine Language section - starting from the absolute basics. This I hope to introduce next issue, but am still looking around for someone to write it on a regular basis. Anyone interested should write to me at 360 Euston Road.

3) Several readers suggested that the newsletter be physically re-formatted - perhaps to an A5 size with binders available. This is being looked into, but I cannot promise anything yet.

4) Those of you who indicated that you would be prepared to talk or write for national magazines on your application, I am passing your names on to our Public Relations people who will contact you as the opportunities arise.

There were of course many other useful points and comments and I shall try to tailor the newsletter to these as far as possible.

RICHARD PAWSON
Editor
Pet Users Club.

Many members have now renewed their subscriptions for next year but for those who haven't and want to carry on receiving the newsletter, please send us a cheque with the form below as soon as possible.

PET USERS CLUB
Membership Renewal Form

Please renew my membership of the Official Pet Users Club for the period 1st JULY 1979 to 30th JUNE 1980

I enclose a cheque/postal order number .....
to the amount of £10.00 (£15.00 for overseas subscriptions) made payable to:-

Commodore Business Machines UK Ltd.

Name: .....

Address: .....

.....

Software Notes

First of all, apologies to you all who are experiencing difficulties in obtaining software from us. We are in the process of integrating our premises for PET: the accounts, distribution and quality control people are all coming to Slough and we are moving to join them.

An improvement may take a couple of weeks or so to be noticeable to you but I shall be personally taking charge of software production and hopefully we will be able to deal with the little problems that hold things up, much more rapidly than we have in the past. Not only should software be easier to obtain henceforth, but quality should be improved.

As a result of all this there are no new releases to announce in this issue of PET Users' News Letter but I can say that we shall soon have a number of new titles from the US, which include some tutorial programs in elementary English, Chemistry and Physics. Meanwhile, developments in the UK are proceeding and I hope to be able to make some interesting announcements in the future.

May I say that as far as I am aware all our current titles run without logical errors of any kind. This is hard to believe but I have not had a logical error in any of our software reported for more than 5 weeks. Are we really that good?

If ever you find a bug in our software please write to us about it and we will, of course, replace the program you have bought. For people that can suggest cures for bugs they find, we are willing to consider a small reward of some kind.

Now a word about the means we are developing for promoting products and services available in this country for the PET.

Firstly, it is our intention to produce a Directory of goods, services and hardware - rather like the Channel Data Book in the US. The categories for entry into our catalogue will be : Software, Hardware and Services. The sort of things we have in mind here are programs written by dealers and indeed, anyone else, who has software available for the PET. Hardware includes any hardware that is appropriate for a PET which could be hung on the IEEE bus, the User Port, the Memory Expansion Port, or indeed an RS232 Interface. Internal add-ons are also of interest. Services would include education and custom programming primarily, but suppliers of furniture and stationery could also come into this category.

If you would like a copy of the forms to fill in, please write to Andrew Goltz at Euston Road.

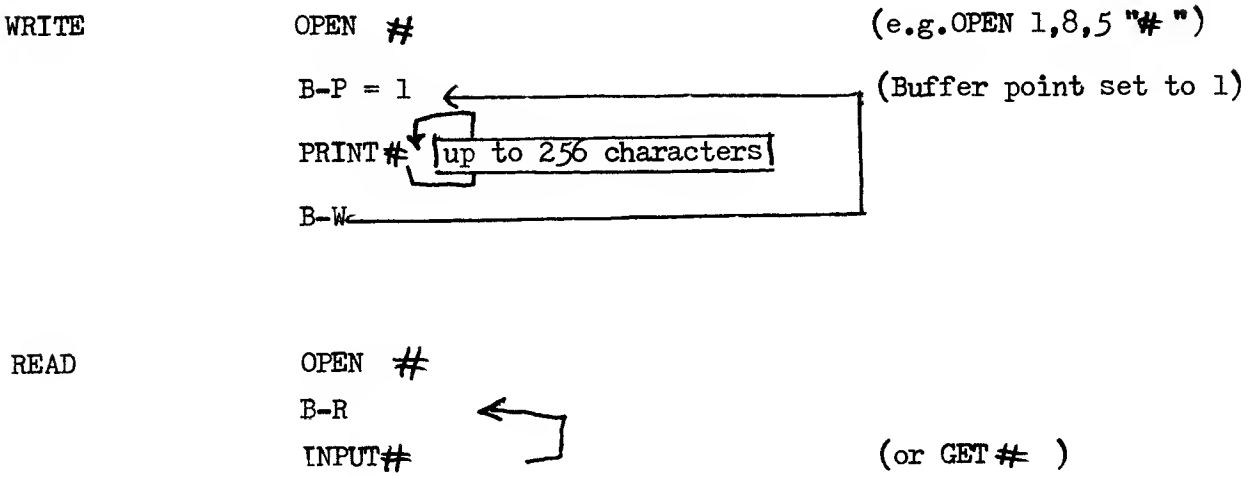
There exist already certain products, both in Hardware and Software, and some services that we at Commodore feel are especially worthy and should be considered for the Commodore Endorsement Scheme. These products will be distinguished by carrying the Commodore logo with the legend "Commodore Approved".

Brief descriptions will be included in our standard literature. For example, as a second section in a leaflet including our software catalogue. A small charge will be made for this service on a royalty basis.

We will be closing the Startrek competition on August 1st, so if you have anything you wish to submit the time is right and I should say that we have so far considered some half a dozen programs, some of which have been extremely good.

Some more remarks about the disk are appropriate whilst the new documentation in preparation in the USA has not arrived.

If you are struggling with direct access, the following 2 diagrams might help:



NOTE:

If GET is used and many strings are held in memory Garbage Collection problems may be encountered. Recurrence assignments of the type A$=A$ + X$ should be avoided (implied by GET# N,A$, say). It is a wiser strategy to use CHR$(13) - carriage return - to delimit fields in records and use INPUTs.

Problems with chaining can be dealt with by re-locating start of variables table with appropriate POKE commands, or where no parameters are to be passed, by arranging for the LOAD command to be executed by printing on the screen and arranging for carriage returns and appropriate cursor controls to be in the keyboard buffer when the calling program halts.

Problems with garbage collection were mentioned last time but I am sorry to say this is more difficult. It is common to all Microsoft basics and is due to the way strings space is organised.

America has started to take software from us, so keep submissions coming, as we can probably tackle international markets with good software - particularly Mathematical, Technical and Scientific stuff.

Lastly, thank you for all your helpful remarks about the problems I mentioned last time with expansion RAM. I am sure you will all be interested to hear that new ROMS cure all of these problems.

Applications

The application for this issue is quite a short one but, since we have so many readers who use their Pets in Education, it should have a fairly widespread interest. This article has been chosen because the author has used some ingenious programming to get a whole lot of information stored in an 8K machine.

+ + + + + + + + + + + + +

My job as a school teacher happens to involve me in organising O and A level examinations in my school, and I decided to see whether I could use my Pet to store the examination entries in any useful way. (This was purely as a challenge to myself - I'm not really making any actual use of the results this year.) The O level exams provided the larger problem; I am dealing this summer with 394 candidates and 25 subjects, and I wanted as a minimum target to be able to devise a program that would output the candidate list for any given subject while holding all the data in store. At the same time, loading the program should not take prohibitively long, and altering entry details should be relatively easy. I have a standard 8K Pet with no 'extras'.

This is of course, a trivial problem if there is plenty of space available, but to fit everything into 8K requires a little trickery. I decided I didn't want to abbreviate candidates' names at all, so I concentrated on storing the subjects for which they were entered as briefly as possible. In the end I used a single integer in the range 1 to 33554431 ($=2^{25} - 1$) for each candidate, the integer being the sum of all the powers of 2 corresponding to the subjects for which the candidate was entered (i.e. 1st subject = 1, 2nd subject = 2, 3rd subject = 4, ..., last subject = 2^{24}). Since we have a good number of candidates only taking one or two subjects, I was able to save a lot of space by ensuring that these subjects came first in the list; thus many candidates' entry code number is only a single digit.

For storage, I thought at first the best plan was going to involve blocking off the top few K of memory and poking my data in byte by byte. This did work all right, but I found it took rather a long time to get all the data in from tape. However, I've found that keeping all my data in data statements works very well. Loading time is not too bad (about a couple of minutes) and as I just want to make one pass right through the data per run there's no disadvantage in not being able to shift the data around easily. Space is still fairly tight, but I can fit in all the surnames (with initials where necessary to distinguish different candidates of the same name) and entries together with a self-explanatory program enabling a listing for any subject to be given. There are still about 700 bytes free, which enables small extra program segments to be written in as required (e.g. to search for candidates who have a timetable clash, or to list all those taking some specified combination of subjects).

I attach a program listing (less most of the data statements), though I dare say this is not of much general interest, being rather tailored to one particular problem.

I am going to try to fit in the exam grades as well - when they become available, but I think that I might run out of memory.

James Clark
5 Park View House,
9A Eastbury Road,
Watford.

EXAM ENTRIES PROGRAM

```
  1  DATANADIMI BHM,1152,NADIMI BRM,1152,DORE,4,TURNER,4,BIGLAND,8
  2  DATACOLBERT,56,DUFFY PC,24,GORDON AC,8,HAGGIS,12,HORSTEAD,12,JOHNSON,24
 etc. etc.
 4Ø  DATACLEMENTS,1115393,FOX,83712,HARTMAN,1Ø5Ø37Ø,HEDDERICK,1115392
etc. etc.
 5Ø  DATAWILLIAMS RD,8475648,BLOM,1Ø5Ø371,ELLIS,832Ø1,FISHER DM,3216384
etc. etc.
 77  DATARAYNAUD,3,STEWART PD,3,STEWART RMF,1,TAYLOR MJ,1,TODD,3,WALTER JF,1
 78  DATAWEBB,3,WILLOUGHBY,3,*********,33554431
1ØØ  PRINT"©↓ ENTER THE CODE NUMBER FOR THE SUBJECT TO BE LISTED:↓↓":T=2Ø
11Ø  PRINT" 1 - FRENCH"TAB(T)"13 - LATIN
12Ø  PRINT" 2 - MATHEMATICS"TAB(T)"14 - GREEK
13Ø  PRINT" 3 - AO ENGLISH"TAB(T)"15 - GERMAN
14Ø  PRINT" 4 - AO GERMAN"TAB(T)"16 - RUSSIAN
15Ø  PRINT" 5 - REL.STUDIES"TAB(T) 17 - ADD.MATHS
16Ø  PRINT" 6 - ART (O&C)"TAB(T)"18 - PHYSICS
17Ø  PRINT" 7 - CHEMISTRY"TAB(T)"19 - PHYS-WITH-CHEM
18Ø  PRINT" 8 - ART (LONDON)"TAB(T)"2Ø - GEN.SCIENCE
19Ø  PRINT" 9 - GEOGRAPHY"TAB(T)"21 - BIOLOGY
2ØØ  PRINT"1Ø - GEOLOGY"TAB(T)"22 - MUSIC
21Ø  PRINT"11 - ENGLISH"TAB(T)"23 - DES.&TECH.
22Ø  PRINT"12 - HISTORY"TAB(T)"24 - AO FRENCH
23Ø  PRINTTAB(T)"25 - OTHERS
24Ø  INPUT"↓↓CODE NUMBER REQUIRED";C
25Ø  IFC<1ORC>25THENPRINT"↓SUBJECT CODE ERROR - PLEASE REPEAT.":GOTO24Ø
26Ø  C=2↑(C-1):PRINT "©";:N=Ø:M=Ø:T=Ø
27Ø  FORI=1to395:READP$,E
28Ø  IF2*INT(E/(2*C))<>INT(E/C)THEN3ØØ
29Ø  NEXT:GOTO35Ø
3ØØ  N=N+1:M=M+1:IFM<21OR(M=21ANDI=395)THENPRINTTAB(T)P$:GOTO29Ø
31Ø  M=1:IFT=ØTHENT=2Ø:PRINT"Ⓗ"TAB(T)P$:GOTO29Ø
32Ø  T=Ø:PRINT"↓PRESS * TO CONTINUE LIST
33Ø  GETZ$:IFZ$<>"*"THEN33Ø
34Ø  PRINT"©"TAB(T)P$:GOTO29Ø
35Ø  PRINT"↓"TAB(T)"TOTAL NO. ="N-1
36Ø  IFT=ØANDM=21THENPRINT"↑
37Ø  PRINT"Ⓗ↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓
```

© = "CLEAR SCREEN"

Ⓗ = "CURSOR HOME"

↑↓ = CURSOR UP, DOWN

"Printout"

1 The SHELL-METZNER sort

Most books on BASIC show how to code a 'ripple' or 'bubble sort. Those of you who have tried it will know that it does it's job accurately but slowly. This is because it 'bulldozes' it's way through the problem, rippling one item to the end of the list on each pass. There are a number of sort routines which are faster for most sets of data. One of them is the Shell-Metzner.

Like the Bubble sort it compares and swops elements. It differs in the way it selects elements for comparision. The underlying theory is complex and perhaps the best way of seeing the sort in action is to use a pack of playing cards, pencil and paper and 'dry run' your way through a sample sort.

If you have the patience to do this you will find that the routine compares elements half the array apart, then elements a quarter of the array apart and so on. Even if you haven't the time to work out how it operates there is no reason why you shouldn't use it in your programs. It is a perfect example of the type of routine where you know what goes in, what comes out, the variables involved and who cares how it works!

It is also a good example of how you can improve your BASIC skills by reading other people's programs. I learnt about this sort by extracting it from a program published in KILOBAUD last year.

As set out here it sorts an alphabetic array held in element one onwards, into ascending order. It is coded as a subroutine and is a perfect candidate for the Templeton merge described in issue 5.

```
59000 REM SHELL-METZNER SORT
59005 N=A: M=A: REM 1 TO A IS ARRAY SIZE.
59010 M=INT(M/2): IF M=0 THEN RETURN
59020 J=1: K=N-M
59030 I=J
59040 L=I+M
59050 IF A$(I)<=A$(L) THEN 59080
59060 F$=A$(I): A$(I)=A$(L): A$(L)=F$: I=I-M: IF I<1 THEN
      59080
59070 GOTO 59040
59080 J=J+1: IF J>K THEN 59010
59090 GOTO 59030
```

Variables used:- A$(),F$,A,I,J,K,L,M,N.

I've not yet tried adapting the routine to deal with element zero. Would someone like to contribute a version? Also I will be delighted to feature other sort routines you let me know about.

2 Layout of BASIC programs.

My youngest daughter, Josephine, is now writing competent programs. 12 years old, she naturally prefers games to more serious applications. Since games programming requires as varied a selection of BASIC facilities as any other applications area, I encourage her, but insist on an orderly approach.

One idea we have both found very useful is to set out our programs in logical blocks. Since our ideas blend with some programming standards recently published in the home computing press, I am prompted to describe the system here.

Line numbers are used to divide the program into logical blocks.

0- 99	Program name, author, when written, latest amendment number and date, references and credits
100- 999	Instructions
1000-39999	Main program blocks
40000-49999	Subroutines particular to this program
50000-	Library subroutines

Each main block and sub-block start on a line which is a multiple of 1000 or 100. The first line is a REM title for the block.

In use, Josie starts her coding by entering these title lines straight off her rough flow chart. The system encourages her to break down the problem into small sections so that at no time does the coding effort required seem overwhelming.

A fragment of one of her programs which illustrates the method is shown below.

```
   10 REM AUNT SALLY
   20 REM JOSIE & MIKE GROSS-NIKLAUS
   30 REM 19/4/79

 1000 REM PRELIMINARIES
 1010 POKE59468,12
 1020 V$="<hms><24curD>"
 1030 PRINT"<cls>"

 2000 REM DRAW FRAME
 2010 PRINTTAB(15)"<rev>AUNT SALLY"
 2020 FORI=1TO9
 2030 PRINTLEFT$(V$,9)TAB((I-1)*3)"<o><#><p><curD>
      <3curL)<%> <'><curD><3curL><L><$><:>"
 2040 NEXT
 2050 PRINTLEFT$(V$,13)" 1  2  3  4  5  6  7  8  9"

 2100 REM INITIAL DELAY
 2110 FORT=1TO1500:NEXT

 2200 REM START TIMING
```

Etc etc.

3 A formatted listing program.

For those who have printers unable to reproduce PET graphics and cursor control characters, the exact listing of BASIC coding can present a problem. The solution is to use a formatting program to convert the unprintable characters into substitute symbols. The game fragment shown in the previous section was listed using such a formatter.

As a first step, remember that you can LIST your coding to a data file on tape using the following direct command sequence.

```
OPEN1,1,1,"<program name>": CMD1: LIST
CLOSE1:CLR
```

The formatter can now read the data, manipulate it in any way you want and print it out.

Andrew Lister of Manchester College of Higher Education and myself have written such a program. It carries out the following formatting:-

Unprintable shifted characters/graphics are printed as the upper case symbol in brackets.

PET cursor and reverse symbols are printed as abbreviations, also in brackets. eg <rev> means reverse on. Where a symbol appears several times in a row, the printout is abbreviated thus...<24 curD>

Line numbers are right aligned.

Where a program line will not fit on the print line, the extension is indented clear of the line numbers.

Where a line starts with a REM statement and the previous one doesn't, a blank line is printed first. This seperates out the line blocks mentioned in the previous section.

The program has options for printers with and without lower case.

You will get a great deal of enjoyment and something really useful if you write such a program for yourself. For those of you too busy to do so, the program can be obtained from me. The price is £15.00p for a fully documented program. Please enclose with your cheque a strong, self-addressed stamped (14p) envelope.

The idea of manipulating BASIC text in this way can be extended to include alterations to the actual coding. In the next issue I will describe how you can write what amounts to a "BASIC SYMBOLIC ASSEMBLER", which will allow you to code 'BASIC' programs with such statements as

```
2020 GOSUB SORT
2030 PAYNET=PAYGRO-PAYDED
```

4 BASIC COMPETITION 3: ARRAY! ARRAY!

Two lists of random numbers, each 255 long, are held in elements 1 to 255 of the single dimension arrays A and B.

Write a BASIC program to sort the two arrays and merge them, so that the lowest number ends up in element A(1) and the highest in element B(255).

The BASIC program which correctly does this in the shortest time and is coded in 30 lines or less will win for the author £10.00p worth of Commodore software.

Lines 1000-1999 are reserved for my test routine. Your program should be written as a subroutine starting at 2000. You may use earlier lines to set up variables and any other arrays you need.

Solutions to me within 21 days of the official publication date of this issue. Please include with your entry the number of jiffies taken to sort your own test arrays.

5 Results of competition 2.

There were 10 entries for the "Long String" competition. Some tried checking the long string for the initial character of the substring before matching the whole substring, others tried a direct match. Much thought had gone into the order in which variables were created in the program, with significant speed improvements. Mr K.D Armstrong of Edinburgh chose line numbers which were multiples of 256, claiming a small speed improvement. (On checking, the improvement was about 2 jiffies per 1000.)

The winner of the £10.00p software prize is Mr James Clark of Watford. His entry processed the 68 occurences of my 10 character substring in 760 jiffies or 12.67 seconds. Mr Clark also took time to produce a search-time table for substrings of various lengths; suggested ways by which his routine could be shortened if the substring length were known and finally suggested how the problem could be made more general (and complex)! The runner-up was R.J. Westmore of Newtyle, Angus with a solution which did my test in 820 jiffies. Congratulations to you both and indeed to all who entered.

```
   1 I=0:L=0
1000 K=255:L=LEN(S$):T$=A$(0):FORP=0TO1020-LSTEP256-L
1010 FORI=1TOLEN(T$)-L+1:IFS$=MID$(T$,I,L)
     THENJ=INT((P+I-1)/K):PRINTJ;P+I-K*J
1020 NEXT:J=INT((P+I-1)/K):T$=MID$(A$(J) P+I-K*J)
1030 IFLEN(T$)<KTHENT$=T$+LEFT$(A$(J+1).K-LEN(T$)))
1040 NEXT:RETURN
```

Mike Gross-Niklaus
25 Longdown Road, Congleton, Cheshire. CW12 4QH.

Peripherals & Attachments

PET TAPE HEAD ALIGNMENT

Up to now - re-alignment of Pet cassette heads to improve 'loading' performance has required time-consuming 'surgery' to dismantle the cassette deck to get at test points, followed by the use of an oscilloscope or other instrument to measure the amplitude of the Record/Play head output signal.

Now Microputers (one of our dealers) have developed a quick and very simple method of checking and rectifying Head Alignment without instrumentation or dismantling of the deck.

The method has been incorporated into a Pet Head Alignment Kit which has been evaluated and approved at our Eaglescliffe factory. The main element of the kit is a piece of 'Firmware' comprising a tape cassette recorded by special techniques on a variable-alignment test rig, and the kit also includes a calibrated head adjustment tool and full instructions.

These kits are expensive to produce (as they cannot be duplicated by conventional means due to the various special recording techniques used in the 14-program test suite) and it is not therefore intended that they would be widely sold in the 'user' market - indeed, it would probably be uneconomic for an individual Pet user to purchase a kit to re-align his tape head on a 1-time basis. However, dealers and service organisations requiring the facility to check cassette decks on a frequent basis would soon recoup the kit purchase price of £25 (+VAT) in engineer time-saving.

For individuals needing only a 1-time head alignment check, for example because of difficulty in loading tapes recorded on other machines, or to equalise the alignment between heads in a 2-cassette system, Microputers will be offering a hire service under which an Alignment kit may be hired for 1 week for £3 (including postage & VAT).

Your contact for Purchase or Hire of Pet Head Alignment Kits is:
Robin Leaver at Microputers, 30 Grange Street, Clifton, Shefford,Beds.
(C.W.O. postal applications only please.)

+ + + + + + + + + +

JOYSTICK UNIT

The Joystick unit reviewed in issue No. 6 of this newsletter was designed by J.A. ORR and G. KNOTT. They have now parted company with DAMS and set up a new company - STACK COMPUTER SERVICES. The Joystick is thus no longer available from DAMS since STACK are now the sole distributers. The units will be available through your local dealer however.

+ + + + + + + + + +

BLINKIN' LIGHTS MACHINE

The sketch overleaf may be of assistance to anyone intending to build the "Blinkin' Lights" machine described in the "User Port Cookbook" operating instructions from Commodore (Order No. MPØ31).

The whole circuit may be built on to a Prototype Circuit Board (RS Components, 488-618) which sits neatly on top of the Pet, held in position by a lug attached to the rear panel top screw, and with the LED's bent forward (as shown in the diagram), enables virtually all of the procedures outlined in the "Cookbook" to be carried out with an immediate visual response, conveniently at eye level. In addition the audio amplifier connection for the shift register modes of CB2 may be accommodated on the board. It has been successfully used to produce the strains of "Auld Lang Syne" (Practical Computing, May 1979).

The LED resistors, of 330Ω, were chosen to limit the current drain, from Pin B-2 of J3, on the Pet's power supply. With the configuration shown the total current drawn is just over 100 mA. Alternatively, Darlington Driver IC's were also tried in place of the DM7404's with a total current drain of about 60 mA for the byte display only. Ten-way ribbon cable and a small DIL Switch (RS337-560) made construction easy, only the layout is somewhat critical, with the solution indicated in the accompanying diagram. The only snags encountered were due to solder running down the pins (cut from the resistor leads) used to terminate the ten-way cable on the board. Single strand wire for jumper leads is essential, otherwise the rails on the board are easily damaged. Only typical jumper connections are shown on the diagram for the sake of clarity.

The construction and operation of the "Blinkin' Lights" machine forms an excellent introduction to understanding the intricacies of the registers of the VIA 6522 and Commodore are to be complimented for making the material available.

Some additional notes:

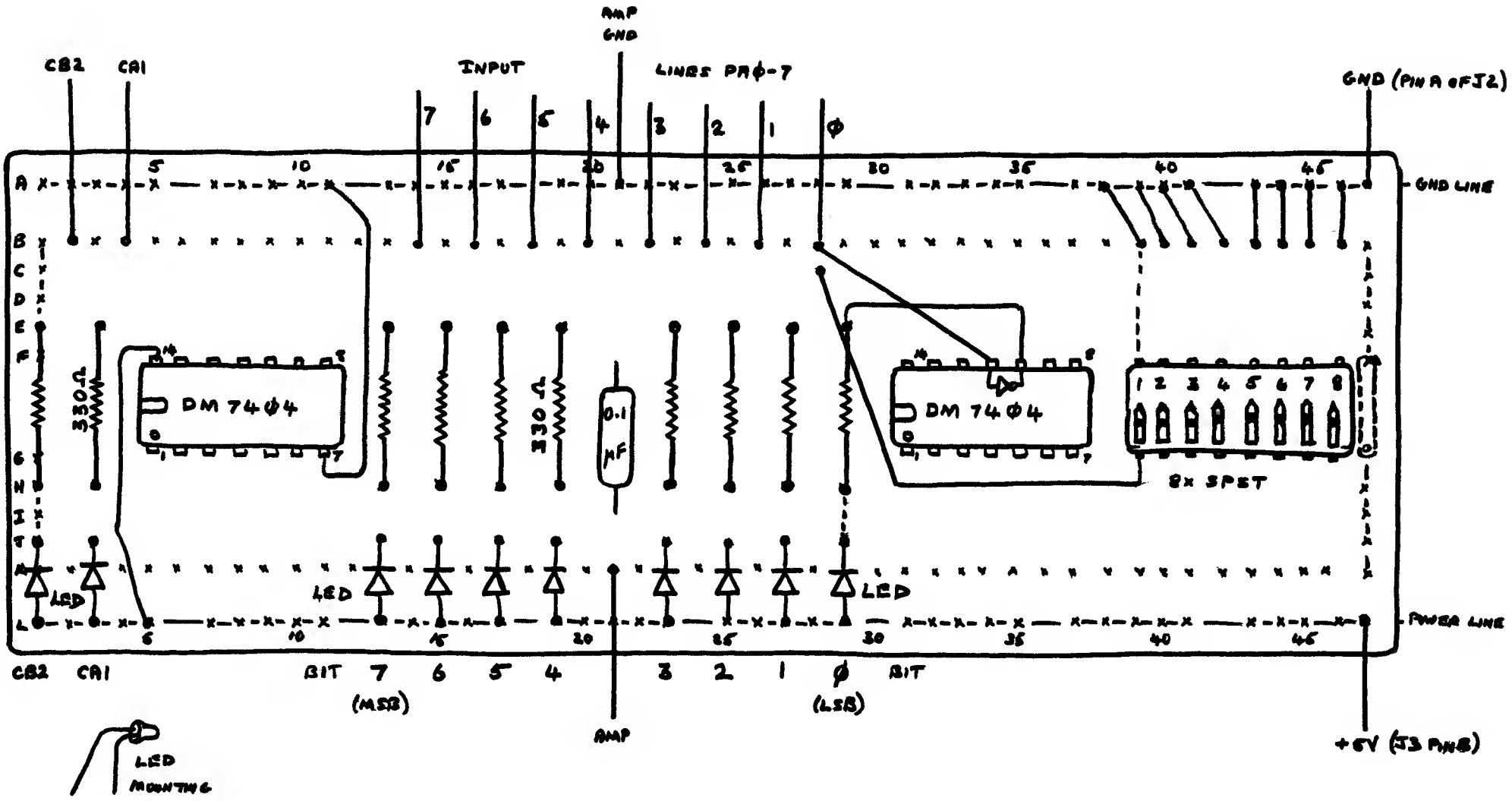
1. The Prototype board contains 47 rows of 5 interconnected contacts, with continuous contact rails top and bottom.

2. The DIL Switches serve to isolate the lines - in which state the output is high with LED's ON. They also serve to ground the PAØ - PA7 lines to the low state with the LED's OFF.

3. A home-made contact of brass strip and contact pin is used in the last row of contacts to momentarily ground the CA1 line.

4. The CB2 line is taken to one of the unused inverters on the 7404#2 and thence by capacitive coupling (0.1 μF) to the amplifier for audio output.

D. MUIR,
Dept. of Physics,
Napier College, Merchiston,
Edinburgh.
Tel: 031.447.7070 Ext.211



BLINKIN LIGHT MACHINE

We are frequently asked for information on how to use the Pet to control a series of instruments for industrial use. The following article comes from Mr. R. Heath of PLESSEY TELECOMMS.LTD. and describes a very professional set up which they currently have in use. We hope that this will answer a lot of your queries.

+ + + + + + + + + + + + +

USING THE IEEE BUS TO DRIVE INSTRUMENTS

In order to use the Pet as a controller when setting up a general purpose audio test set, the standard routines described below were developed.

The basic instruments controlled were:-

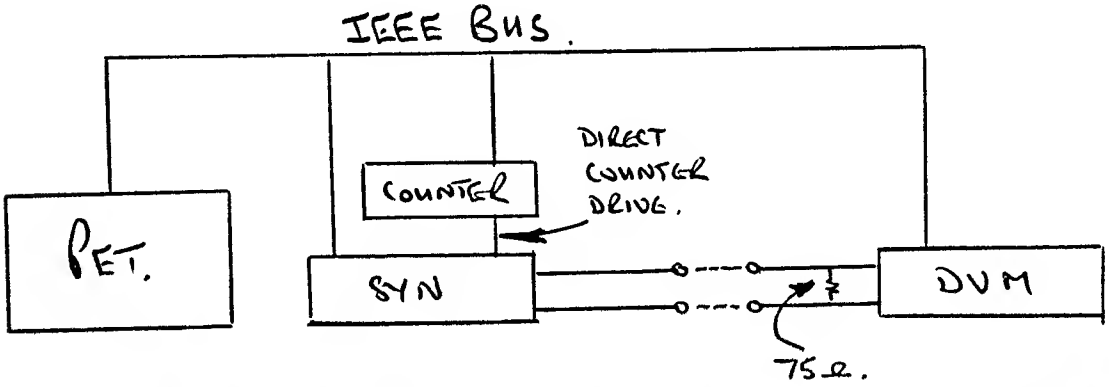
1. Frequency Synthesiser. Hewlett-Packard 3320B.
2. Digital Voltmeter. Hewlett-Packard 3455A.
3. Frequency counter. Hewlett-Packard 5328.

This program illustrates the operation of the three instruments under the control of the Pet.

The synthesiser output is connected directly to the D.V.M. for illustration purposes and readings are displayed by the Pet over the required Frequency range and at the various synthesiser output levels.

The interconnections for the set up are as shown, the IEEE ports of all instruments and the Pet are parallel connected.

The frequency counter input is driven directly from the Synthesiser high level output.



As written, the conversion of volts to dBs assumes 75ohms circuits are being tested, a 75ohms resistor must therefore be connected across the D.V.M. input as shown.

At each change of frequency or level it is necessary to allow a settling time for the instruments and the device being tested.

This time is variable depending upon the degree of Frequency or level change.

It is usually less than 0.2 seconds, but a wait of 1 second had been assumed in all cases for illustration.

Lines 1 to 70 describe what the program can do.

The instruments talk and listen through Files which need to be opened and then closed at the end of each test run.

Lines 190 to 198 open the required Files starting at File 4.

Note that Files 1, 2 and 3 are reserved for tape# 1, tape# 2 and screen respectively.

The select codes required by the Pet must be in decimal, therefore the H.P. ASCII codes need to be converted as follows:

SYNTHESISER LISTEN.	ASCII 3 = DEC 19 + 32
D.V.M. LISTEN.	ASCII 6 = DEC 22 + 32
D.V.M. TALK.	ASCII V = DEC 22 + 64
COUNTER LISTEN.	ASCII 9 = DEC 25 + 32
COUNTER TALK.	ASCII Y = DEC 25 + 64

Lines 202 to 232 set up the operating conditions for each instrument in turn, using the Print command to the relevant instrument File.

The characters used are peculiar to each instrument and a full description of their meaning can be obtained from the relevant handbooks.

The meaning of the characters used in this program are as follows:

FREQUENCY SYNTHESISER

A+Ø = ØdBm

F8ØØ = 8ØØHz

R2 = Range 2

M = Vernier out = disable Fine Frequency control

D3 = Delay code 3

C = command (initiates only)

DIGITAL VOLTMETER

F2 = Function, AC volts

R2 = Range 7, auto

A1 = Auto cal., off

H1 = High resolution, on

M3 = Maths, off

T1 = Trigger, internal

FREQUENCY COUNTER

P = Remote program initialise

F4 = Frequency, A input

G6 = Frequency resolution, 1 Hz

S13 = Multiple measurement, no service request

T = Reset and trigger

Line 401 sets up the loop to change the level from -10 to +10dB in steps.

Line 410 sets the synthesiser level to the value of L and D2,C triggers the change.

Lines 425 to 440 calculate the synthesiser frequency (F).

Line 450 calls up sub 5000 which sets synthesiser Frequency.

Values for Frequency (F1), Frequency range setting (S) and amplitude levelling (D1) are obtained.

These values are then sent to the synthesiser using the Print# 4 command in line 5070.

The C in this line is the trigger command.

Line 460 sends a "comma" trigger command to the D.V.M. and loops until a new reading is obtained through input File 5.

Line 470 obtains a Frequency reading in a similar way.

Line 490 converts the voltage reading into dBm and line 500 displays the results on the V.D.U.

The program can be easily modified to allow tests to be done over any specific range of frequency and level within the limits of the test instruments.

Whilst other instruments could be substituted for those described, some alteration in setting up and trigger commands would be necessary.

The necessary alterations can usually be deduced from the relevant handbooks. However, it must be remembered that all instruments are not triggered in the same way.

+ + + + + + + + + +

DERIVATION OF INSTRUMENT ADDRESS

HEWLETT-PACKARD DVM (3455A)

		BINARY	PET DECIMAL	HEWLETT-PACKARD ASCII
Instrument Address		10110	22	
To Listen	01	10110	54 = 22 + 32	6
To Talk	10	10110	86 = 22 + 64	V

INSTRUMENT ADDRESS SWITCHES

Address Bits 6 and 7 are set by the controller.

01 For LISTEN. PRINT#22,

10 For TALK INPUT#22,

The above is typical of most Hewlett-Packard instruments.

PRINT OUT OF RESULTS.

SEND LEV(DBM)	SEND FREQ(HZ)	REC. LEV(DBM)	REC. FREQ(HZ)
-10	20	-9.97	19
-10	40	-9.98	39
-10	60	-9.98	59
-10	80	-9.98	80
-10	100	-9.98	99
-10	200	-9.98	199
-10	400	-9.98	399
-10	600	-9.98	600
-10	800	-9.99	800
-10	1000	-9.99	1000
-10	2000	-9.98	2000
-10	4000	-9.98	3999
-10	6000	-9.98	6000
-10	8000	-9.98	8000
-10	10000	-9.98	9999
-10	20000	-9.99	19999
-10	40000	-10.02	39999
-10	60000	-10.04	59999
-10	80000	-10.07	79999
-10	100000	-10.08	99999
-10	200000	-10.09	199302
-10	400000	-10.08	399999
-10	600000	-10.89	599999
-10	800000	-10.89	799999
-10	1000000	-11.97	999998
0	20	.03	19
0	40	.02	40
0	60	.02	60
0	80	.02	79
0	100	.02	99
0	200	.01	199
0	400	.01	399
0	600	.01	600
0	800	.01	799
0	1000	.02	999
0	2000	.02	1999
0	4000	.02	4000
0	6000	.02	5999
0	8000	.02	7999
0	10000	.02	9999
0	20000	.01	19999
0	40000	.01	39999
0	60000	.02	59999
0	80000	.02	79999
0	100000	.04	99999
0	200000	.18	199999
0	400000	.52	399999
0	600000	.03	599999
0	800000	.07	799998
0	1000000	-.14	999998
10	20	10.02	19
10	40	10.02	39
10	60	10.01	60
10	80	10.01	80
10	100	10.01	100
10	200	10.01	200
10	400	10.01	399
10	600	10.01	599

```
1 REM  PET104  'AUDIO TEST'
10 PRINT'PROGRAM TO DEMONSTRATE THE IEEE-488 BUS':PRINT
14 PRINT'SYSTEM DRIVING A GENERAL PURPOSE AUDIO ':PRINT
18 PRINT'TEST SET.'
22 FORI=1TO40:PRINT'-';:NEXT:PRINT
26 PRINT'THE INSTRUMENTS USED ARE:-':PRINT
30 PRINT'HEWLETT PACKARD 3320B SYNTHESIZER
34 PRINT'HEWLETT PACKARD 3455A DIGITAL VOLTMETER'
38 PRINT'HEWLETT PACKARD 5328A COUNTER':PRINT
42 PRINT'THE PROGRAM CAN BE USED AS A BASIS FOR
46 PRINT'A SETUP CAPABLE OF DOING MEASUREMENTS
50 PRINT'OVER THE FREQUENCY RANGE 20HZ-20KHZ.'
54 PRINT'IT MAY ALSO BE USED FOR TAKING HIGHER
58 PRINT'FREQUENCY MEASUREMENTS(UP TO 1MHZ)
62 PRINT'PROVIDING THE USE OF THE DVM AS A LEVEL MEASURING DEVICE IS ADEQUA
66 PRINT
67 PRINT'     PRESS A KEY TO CONTINUE.
68 GETK$:IF K$=''GOTO68
69 PRINT''
70 PRINT'TERMINATE THE DVM IN 75 OHMS.
74 PRINT'CONNECT THE SYNTHESIZER OUTPUT TO THE
78 PRINT'DVM AND COUNTER INPUTS.':PRINT
82 PRINT'THE PROGRAM CONVERSION OF VOLTS TO DBM.
86 PRINT'ASSUMES A 75 OHMS TERMINATION.
90 PRINT'FOR 600 OHMS, CHANGE 0.27386 IN LINE
94 PRINT'490 TO 0.77459.
98 PRINT'THE WAIT TIMES USED REPRESENT 1 SECOND.
102 PRINT'THESE CAN BE CHANGED ACCORDING TO THE
106 PRINT'MEASURING CIRCUMSTANCES.':PRINT
110 PRINT'A TYPICAL PRINT OUT OF RESULTS IS AS    FOLLOWS:-':PRINT
118 PRINT' SEND      SEND      REC.       REC.
122 PRINT'LEV(DBM)  FREQ(HZ) LEV(DBM)   FREQ(HZ)
126 PRINT' -10        20        -9.76      19
130 PRINT' -10        40        -9.77      39
134 PRINT' -10        60        -9.77      60
136 PRINTTAB(15)'ETC.':PRINT
140 PRINT'     PRESS A KEY TO CONTINUE.
144 GET K$:IF K$=''GOTO144
148 PRINT'THE MEASURING RANGE OF THE PROGRAM AS'
152 PRINT'WRITTEN IS:-':PRINT
156 PRINT'    LEVEL          -10DBM TO +10DBM':PRINT
160 PRINT'    FREQUENCY       20HZ TO 1MHZ':PRINT
164 FORI=1TO6:PRINT:NEXT
170 PRINT'    PRESS A KEY TO START MEASURING.
174 GET A$:IFA$=''GOTO174
176 PRINT'MEASURING
180 OPEN9,4:REM PRINTER LISTEN--SELECT CODE 4
190 OPEN4,19:REM SYN.(LISTEN)
192 OPEN5,22:REM DVM.(LISTEN OR TALK)
196 OPEN6,25:REM CTR.(LISTEN OR TALK)
200 REM SET UP INSTRUMENT OPERATING        CONDITIONS
202 PRINT#4,'A+0,F800,R2,M,D3,C':FORY=1TO1000:NEXT
212 PRINT#5,'F2R7A1H1M3T1':FORY=1TO1000:NEXT
232 PRINT#6,'PF4G6S13T':FORY=1TO1000:NEXT
250 PRINT#9,'                  FULL PRINT OUT OF RESULTS.':PRINT#9
300 PRINT#9,'    SEND              SEND             REC.            REC.'
310 PRINT#9,'   LEV(DBM)         FREQ(HZ)        LEV(DBM)       FREQ(HZ)'
400 REM FOR LEVEL -10 TO +10 STEP 10DB
401 FORL=-1000TO1000STEP1000
```

```
410 PRINT$4,"A"L",D2,C":FORY=1TO1000:NEXT
420 REM FOR FREQUENCY 20HZ TO 1MHZ,        VALUE SET BY N AND DECADE SET BY P
425 FORP=1TO5
430 FORN=2TO10STEP2
440 F=INT(N*10^P)
450 GOSUB5000
455 REM TRIGGER INSTRUMENTS AND OBTAIN READING
460 A$="*":PRINT$5,",":INPUT$5,A$:IFA$="*"GOTO460
470 B$="*":INPUT$6,B$:IFB$="*"GOTO470
480 B=VAL(B$)
485 REM CONVERT VOLTS TO DBM
490 A=VAL(A$):A=20*(LOG(A/.27386)/LOG(10)):A=(INT(A*100))/100
492 L1$="                "
494 L$=STR$(L/100):L$=L$+L1$:L$=LEFT$(L$,16)
496 F$=STR$(F):F$=F$+L1$:F$=LEFT$(F$,16)
498 A$=STR$(A):A$=A$+L1$:A$=LEFT$(A$,16)
500 B$=STR$(B):B$=B$+L1$:B$=LEFT$(B$,16)
510 PRINT#9,"      "L$,F$,A$,B$
580 NEXTN
590 NEXTP
600 NEXTL
900 CLOSE4:CLOSE5:CLOSE6:CLOSE9
1000 END
5000 REM SET SYNTHESIZER FREQUENCY
5001 REM F1=FREQUENCY
5002 REM S=FREQUENCY RANGE SETTING
5003 REM D1= AMPLITUDE LEVELING
5010 D1=0
5020 IF F>1299000 THENS=6:F1=F/10000:GOTO5070
5030 IF F>129900 THENS=5:F1=F/1000:GOTO5070
5040 IF F>12990 THENS=4:F1=F/100:GOTO5070
5050 IF F>1290 THENS=3:F1=F/10:GOTO5070
5060 F1=F:S=2:D1=3
5070 PRINT#4,"F"F1",R"S",D"D1",C":FORY=1TO1000:NEXT
5080 RETURN
READY.
```

HIGH SPEED TAPE CONTROL

By W M McCRACKEN

I was very interested to read the article on "HIGH SPEED FILE ACCESS" in the Commodore Pet Users Club News Letter, Issue 4(1). This, together with an article by D H Wilcox in the American PET User Notes(2) prompted me to investigate the possibility of developing a fairly general fast access method.

I have since done some work on this project and include my findings.

The NEED for HIGH SPEED ACCESS

When developing different programs for the PET it very soon becomes apparent that a great deal of time is spent in "finding" a particular program on tape, (or space to save a program).

This is due to the fact that, since there is no counter on the tape recorder, we cannot accurately position the tape at the start or end of any particular file.

We can "estimate" these positions using the graduations on the cassette window, (a very approximate method). We can Fast Forward or Rewind during the running of the LOAD or VERIFY programs (again somewhat hit or miss).

This latter method, however, is the key to a considerably faster search. If we could control the Fast Forward run and know in advance the time to each file start, then we could speed up software development.

The FAST FORWARD SPEED

As pointed out in (1), "Different types of tape and different PETs Fast Forward at slightly different speeds", and as we have all noticed or heard, the Fast Forward speed increases as the tape winds on.

These statements suggest that it is difficult to get any general rules for Fast Forwarding to discrete positions on the tape. This was the first

problem I decided to tackle.

I wanted to know just how the Fast Forward time varied, so I produced a tape, "marked off", with evenly spaced timing numbers. This was accomplished using PROGRAM "PRINT ON TAPE" and a C60 cassette, (I usually use C60's).

PROGRAM "PRINT ON TAPE"

```
10 POKE243,122:POKE244,2:OPEN1,1,1
20 FORZ=100TO416
30 PRINTZ:REM**PRINT ON SCREEN
40 PRINT#1,Z:REM**PRINT ON TAPE
50 P=PEEK(625):IFP>175THENPOKE59411,53
60 IFP<189GOTO40
70 POKE625,191:NEXTZ
80 CLOSE1
```

Line 10 checks the number of characters in the tape 1 buffer and if this is greater than 175, turns on the built in cassette motor. Line 60 again checks the buffer and if over 189, shifts control to line 70 where the buffer size is made 191, thus forcing the buffer contents on to the tape, the FOR-NEXT loop is then continued.

Once the timing tape had been produced I used the following PROGRAM, "TAPE READ" in order to see at which number the tape stopped, with varying Fast Forward times.

PROGRAM "TAPE READ"

```
10 PRINT"
20 FORI=STO100STEP5:GOSUB200
30 OPEN1:GOSUB200
40 GOSUB300
50 PRINT"■":PRINT"WITH A FAST FORWARD TIME OF";I;"SECONDS"
60 PRINT:PRINT"THE NUMBER ON THE TAPE #1 IS"..
70 FORZ=1TO10:GET#1,A$:PRINTA$;:NEXTZ:CLOSE1:NEXTI
80 END
200 PRINTSPC(120):PRINT"PRESS RFWIND ON TAPE #1
210 IFPEEK(59411)<>53GOTO210
220 PRINT:PRINT"PRESS STOP ON TAPE #1 WHEN FULLY REWOUND
230 IFPEEK(59411)<>61GOTO230
240 PRINT"■":RETURN
300 PRINT:PRINT"PRESS FAST FORWARD ON TAPE #1
310 IFPEEK(59411)<>53GOTO310
320 TF=TI+60*I
330 IFTI<TFGOTO330
340 POKE519,52:POKE59411,61
350 PRINT:PRINT"PRESS STOP THEN PLAY ON TAPE #1
360 IFPEEK(59411)<>53GOTO360
370 RETURN
```

Line 30 - OPENS the tape channel for read

Line 70 - accepts single characters from the tape and displays them on the screen, then closes the channel.

SUB 200 - makes sure that the tape is fully rewound, and that all cassette keys are up, the key sensing being done by lines 210 and 230 . Notice that the inequality sign has been used (ie<>) rather than =. This was done because when = was used, basic sometimes passed this control, when no switching had taken place. This was not due to switch bounce since control was held at line 210 or 230 for times of 7+ seconds. This suggests that the value of byte 59411 was at some times set to a value other than 53 or 61. I have checked this using other PETs and have found that indeed the = is unreliable. Does anyone know why this should be or is it a fault.

sub 300 - Fast Forwards for the desired number of seconds using the built in clock.

TABLE 1

FAST FORWARD TIME (SEC)	5	10	15	20	25	30	35	40	45	50
NUMBER ON TAPE	109	120	131	143	156	170	184	214	214	230
FAST FORWARD TIME (SEC)	55	60	65	70	75	80	85	90	95	-
NUMBER ON TAPE	247	264	282	301	320	340	361	382	405	-

This was repeated many times in order to estimate the fluctuation of tape speed on Fast Forward, especially due to running cold or warm. These repeats were somewhat shorter tests, being done for times of 10, 30, 50, etc. seconds, the ranges of numbers being as shown.

TABLE 2

FAST FORWARD TIME (sec)	10	30	50	70	90
RANGE OF NUMBERS	119-120	170-170	230-231	301-302	282-284

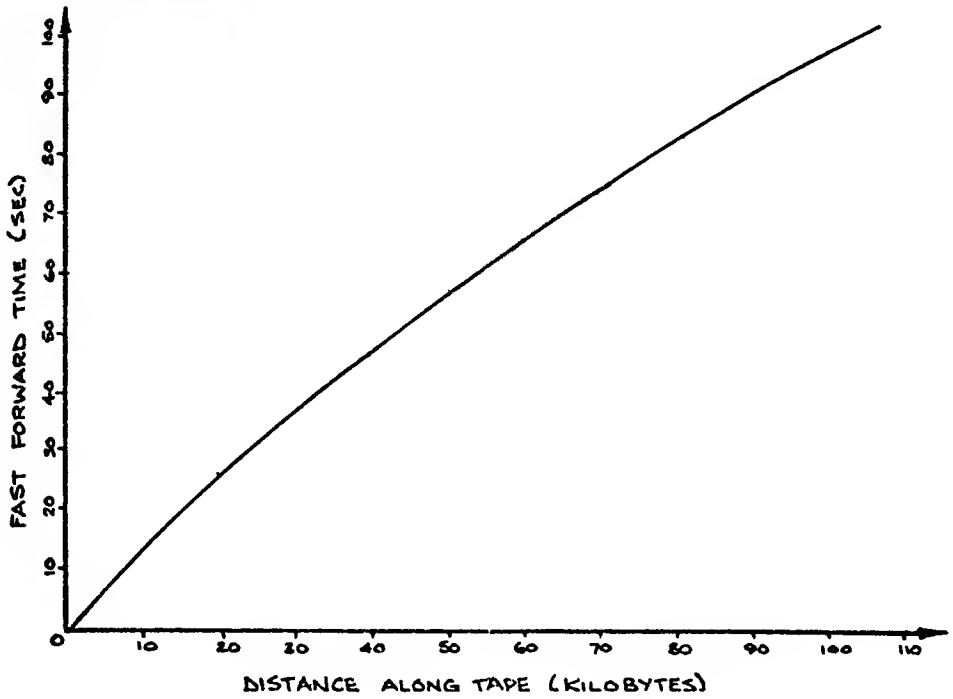
It must be remembered that each number spanned a block on the tape, the length of which was approx 5.75 sec at Play Speed. This time together with other important ones was estimated using a stop watch, or the PET's clock,

eg Time to save ø Bytes = 15.8 sec
Time to save 2811 Bytes = 66.35 sec
Time to save 5355 Bytes = 112 sec
Time to save 6678 Bytes = 135 sec

From this we can get the Time/Byte relationship eg
(135 -15.8) / 6678 = 0.018

The total PLAY time less the 7 seconds lead was measured as 31.44 minutes. This means that we are capable of storing 31.44 / 0.018 = 104800 Bytes on each side of the tape. Notice that the time for OPEN and CLOSE file was accounted for by the SAVE ø Bytes time. All this information can be used to change the timing blocks into numbers of bytes.

We can now represent TABLE 1 as a graph of Fast Forward Time v Distance along tape (kilobytes).



The slope of this curve approximately halves from start to finish as we move along the X axis. This means that, as we approach the end of the tape on a Fast Forward run, the speed of the tape is TWICE what it was at the start. This is quite significant.

The way in which this information was received was not convenient to work with, so I used a "curve fitting" package to determine the coefficients of a polynomial which would give a good fit to this set of points. It turned out that a third order satisfied these demands.

Armed with this equation, if we know the distance along the tape, in bytes, to any desired point we can calculate the corresponding Fast Forward time. However TABLE 2 shows that there are some fluctuations in these times and therefore prohibits a really accurate positioning.

In order that these fluctuations should have little effect it was decided that the tape should be thought of as being divided into blocks, each block being able to accept one COMPLETE memory, ie approx 7100 Bytes, plus 900 to allow for OPEN and CLOSE. This means that each side of a C60 tape can be divided into 13 equal spaces, (ie 13 programs / sides).

The fact that each tape cassette has its own dynamic characteristics, can be allowed for by proportionally altering the polynomial coefficients. This assumes that each cassette's characteristics follow, fairly closely, the same pattern, a reasonable assumption.

If we therefore use the same PET with different cassettes all we have to do is establish the total Fast Forward time for each cassette and alter one line in the control program, to "tune" the system.

We can extend this thinking to allow for different PETS. All we have to do for this is to establish the ratio, PLAY time / Fast Forward time for any <u>one</u> cassette. This gives us a constant for this PET.

These ideas have been put together and are presented here as a "contents" program.

THE CONTENTS PROGRAM

I must emphasise that the reader does not have to follow all that has gone previously. The following information will give the reader a working

contents program.

```
40 PC=5.073E-2:REM**TO BE ALTERED FOR EACH PET
50 FFT=98:REM**TO BE ALTERED FOR EACH CASSETTE
60 GOSUB170:PRINT"PRESS STOP ON TAPE #1
70 IFPEEK(519)<>0GOTO70
110 GOSUB170
130 GOSUB190
140 GOSUB430
150 GOSUB480
160 END
170 REM**CLEAR SCREEN
180 PRINT"■";:RETURN
190 PRINT"          ** CONTENTS  PROGRAM **
200 PRINT:PRINT"    ┌──────┬──────────┬──────┬──────────┐
210 PRINT"    |CODE | PROGRAM  | CODE | PROGRAM  |
220 PRINT"    |──────┼──────────┼──────┼──────────|
230 PRINT"    |  -   | CONTENTS |   G  |          |
240 PRINT"    |      |          |      |          |
250 PRINT"    |  A   |          |   H  |          |
260 PRINT"    |      |          |      |          |
270 PRINT"    |  B   |          |   I  |          |
280 PRINT"    |      |          |      |          |
290 PRINT"    |  C   |          |   J  |          |
300 PRINT"    |      |          |      |          |
310 PRINT"    |  D   |          |   K  |          |
320 PRINT"    |      |          |      |          |
330 PRINT"    |  E   |          |   L  |          |
340 PRINT"    |      |          |      |          |
350 PRINT"    |  F   |          |      |          |
360 PRINT"    └──────┴──────────┴──────┴──────────┘
370 PRINT:PRINT"     SELECT APPROPRIATE CODE FOR
380 PRINT"
390 PRINT"     POSITION ON TAPE
400 GETC$:IFC$=""GOTO400
410 IFC$<"A"ORC$>"L"GOTO400
420 BS=ASC(C )-64:RETURN
430 REM**CALC FAST FORWARD SPEED (SECONDS)
440 BS=BS*8000
450 FT=.11594E1+.13985E-2*BS-.71234E-8*BS↑2+.24540E-13*BS↑3-3.5562
460 FT=FT*FFT/98*19.7134*PC
470 RETURN
480 REM**TAPE CONTROL
490 GOSUB170:PRINT"PRESS FAST FORWARD ON TAPE #1
500 IFPEEK(59411)<>53GOTO500
510 FT=TI+FT*60
520 IFTI<FTGOTO520
530 POKE519,52:POKE59411,61
540 GOSUB170:PRINT"PRESS STOP ON TAPE #1
550 IFPEEK(519)<>0GOTO550
560 GOSUB170:PRINT"TAPE IS NOW IN CORRECT POSITION
570 PRINT:PRINT:PRINT"YOU MAY 'LOAD' OR 'SAVE ' YOUR PROGRAM"
580 PRINT:PRINT"AS NORMAL
590 RETURN
```

Line 40 - If this is the FIRST TIME the program has been run on your PET, then for any C60 cassette, determine the ratio Fast Forward time (sec) / Play Time (sec). This is PC in the program, and should be altered accordingly. This need only be done "ONCE" for any one PET.

Line 50 - Determine the Fast Forward time in seconds for the particular cassette to be used.

This is FFT in program, and should be altered accordingly. This need only be done "ONCE" for any one cassette.

Line 70 - Makes sure that all the keys on TAPE 1 are "up".

Line 190 - 470 Prints contents table and waits for code A through L, then calculates the corresponding number of blocks along the tape (BS).
Line 440 converts this number into bytes.
Line 450 calculates the Fast Forward time in seconds, using the polynomial, and allows for the length of the contents program itself (ie 3.5562 secs).
Line 460 takes account of different PET's and cassettes.

Line 480 - 590 Fast forwards the tape to the correct position (similar to PROGRAM "TAPE READ" as described earlier.

HOW TO USE THE PROGRAM

1. SAVE and VERIFY contents program on tape 1
2. Rewind tape#1 and LOAD/RUN
3. Tape will now be in desired position
4. Type NEW and enter your own program
5. SAVE and VERIFY your new program on tape 1
6. Update contents table

TO LOAD from this tape do steps 2 & 3 then LOAD as normal.

Note: Step 1. need only be done once for each side of the cassette.

It can be argued that lines 40, 50 and 460 can be removed from the program to make it less bother. This may be so, but the condition can then arise, due to the combination of PET and cassette, that the "block" length on the tape is considerably less than one "COMPLETE" memory. This could possibly cause "rub-out" problems.

We could eliminate this problem by making the "block" longer, (say 10 programs per side instead of 13), but this may cause longer "SEARCHING" times when using different PETs.

Finally, since I have the use of 2 PETs I have written in the following lines which allow me to use either of the PETs without worry.

```
80 GOSUB170
90 PRINT"WHICH PET ARE YOU USING ? NO.1 OR NO.2
100 GETA:IFA=0GOTO100
120 IFA=2THENPC=107/2060
```

Conclusions

I have been using this program myself and have found that it considerably shortens the searching times associated with LOAD and SAVE commands.

It also eliminates one problem associated with file naming, that is, all programs on the one tape could have exactly the same name, since the program converts these to CODES.

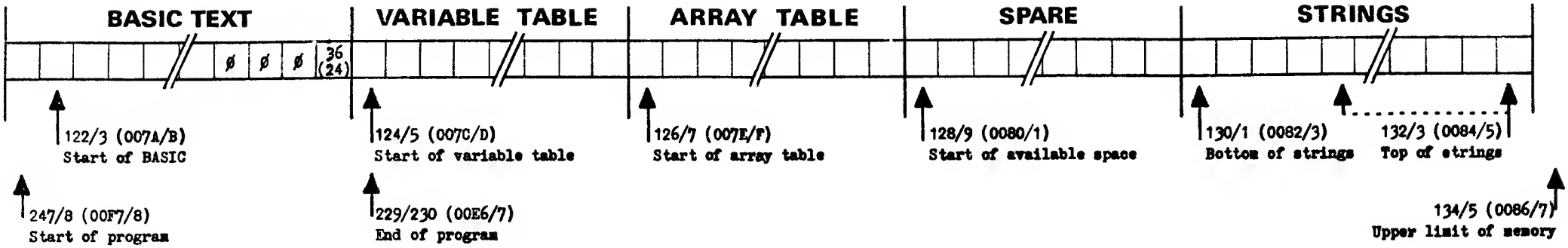
+ + + + + + + + + +

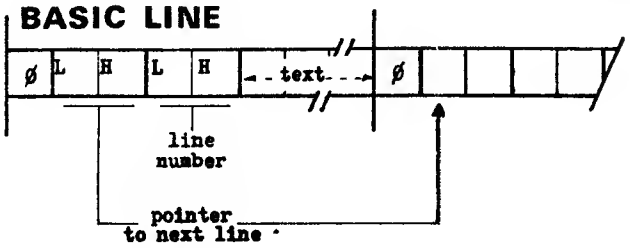
We are greatly indebted to the author for this excellent article. Mr. McCRACKEN comes from the Department of Mechanical Engineering, Design & Drawing in the University of Strathclyde - a very PET orientated University!

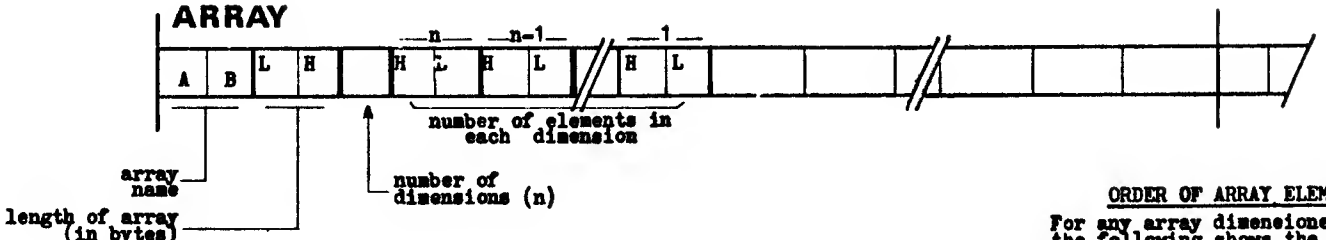
+ + + + + + + + + + + +

Mr. R. TODD has sent in a very helpful chart which summarises the Pet's use of memory for programs and variables - in a quick reference graphical form, and is printed overleaf.

+ + + + + + + + + + + +



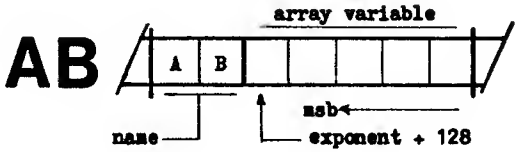


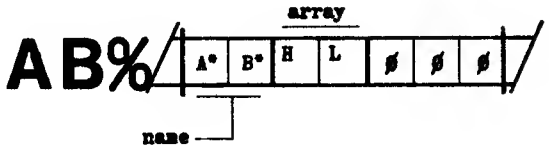


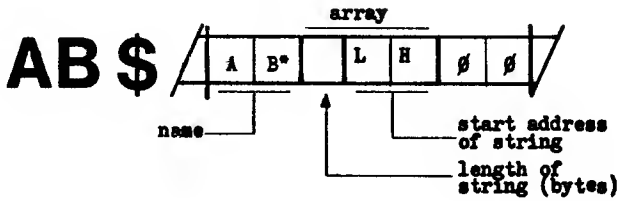
ORDER OF ARRAY ELEMENTS

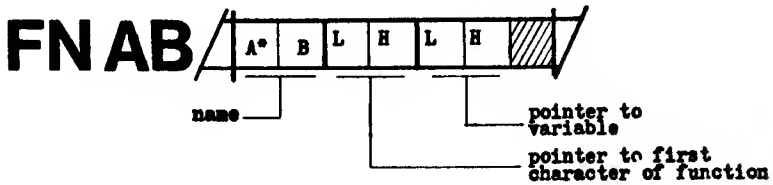
For any array dimensioned to (2,2,2) the following shows the order in which the array elements are placed in memory.

0, 0, 0
1, 0, 0
2, 0, 0
0, 1, 0
1, 1, 0
2, 1, 0
0, 2, 0
1, 2, 0
2, 2, 0
0, 0, 1
1, 0, 1
and so on...
to 2, 2, 2









PET REFERENCE SHEET 7: Memory allocation for BASIC program and variables

NOTES: All values are in decimal - those in brackets are in HEX
* indicates that bit 7 is set to 1
The array name takes the same form as the name for the ordinary variable
Sections of variables indicated as 'array' are those sections which are included in array tables

We have to make an apology to those users who tried our tip for speeding up the listing with POKE 59490,60 and got some very peculiar results. It has since been found that this POKE has all sorts of nasty side effects and we therefore recommend that you don't attempt it.

+ + + + + + + + + +

Mr. M. Bennet has suggested a way of greatly improving the routine for copying a screen into 4 strings in Julian Allason's article in the last issue:

```
5020  PK = PK - (PK AND 128) + (PK AND 64) - 2*(PK AND 32) + 64
```

Lines 5022 to 5028 can then be deleted.

This routine could be incorporated as a function if used frequently.

+ + + + + + + + + +

Finally, I have adapted a routine first sent in by Dr. I.C. SMITH of Queen Elizabeth college, London for plotting the distribution of a variable. In his application, the variable (x) was read from a device on the 8-bit User port although it may be generated inside a program. Entered as it is printed here, the routine is being used to test (visually) the RND function on the PET.

```
 1   DIM D(30) : LS = 33574
 2   PRINT "Clear Home"
10   x = INT(RND(.5)*30 + 1)
20   D(x) = D(x) + 1
30   LC = INT(D(x) - 8* INT(D(x)/8))
40   L = INT((D(x) - 1)/8)
50   IF LC=0 THEN DC = 160
60   IF LC = 1 THEN DC = 100
70   IF LC = 2 THEN DC = 111
80   IF LC = 3 THEN DC = 121
90   IF LC = 4 THEN DC = 98
100  IF LC = 5 THEN DC = 248
110  IF LC = 6 THEN DC = 247
120  IF LC = 7 THEN DC = 227
130  POKE LS - 40* L  + x,DC
140  GOTO 10
```

Notes:

1. The IF.....THEN statements could be removed by means of a dimensioned variable holding the DC constants but are shown here to illustrate the operation of the routine.

2. LS is the location of the bottom left hand corner of the graph. It has been set here at 33574 to allow room for axis and labelling.

3. Any function could be put in place of line 10 .